1

Update Policy of Dense Maps:

efficient algorithms and sparse representation

Manuel Yguel, Olivier Aycard, and Christian Laugier

ε-motion, GRAVIR-UJF-INRIA-INP Grenoble, France firstname.lastname@inrialpes.fr

1.1 Introduction and Previous Work

Providing a robot with a fully detailed map is one appealing key for the Simultaneous Localisation and Mapping (SLAM) problem. It gives the robot a lot of hints to solve either the data association or the localisation problem itself. The more details are in the map, the more chances are that different places may appear differently, solving ambiguities. The more landmarks are used, the more accurate are the algorithms that solve the localisation problem since in a least square sense an approximation of the solution is more precise. Last, it helps a lot in the presence of a few dynamic objects because these moving parts of the environment remain marginal in the amount of data used to model the map and can thus be filtered out. For instance, the moving objects can be detected or cancelled in the localisation procedure by robust techniques using Monte-Carlo algorithms [6] or RANSAC [4].
The environment models that provide such possibilities are named *dense maps*. To build such a representation from range-finders there exist in the literature a lot of propositions: using the point clouds generated by the sensors [2], using grid-based representation [3], [5] or using more geometrical modelling involving feature fitting such as lines or planes [14], [13]. From our point of view the second kind of representation using occupancy grid (OG) is of particular interest since it offers a strong mathematical framework for updates or merges of maps even if the measurements or the map disagree. For example this property is useful if a part of the map, that was not observed, had changed a lot like the streets of a city where the cars park and go.
However as their name suggests it dense maps comes with a memory burden. For instance raw scan records of 361 points of a laser range-finder operating at $50Hz$ produce in 3 hours almost 195 millions of points which make $780Mb$ of data and a grid-based representation that use cubic cells with a side of $5cm$ needs 3.6 billions of cells to cover an area of $3km^2$ which make $28.8Gb$ for an (OG). Naturally this is a hard limitation for such maps and that is why the design of sparse representations is very important.
In this paper we present a review of update methods for grid-based maps like OGs under the requirement of a sparse representation emphasizing the pros and the cons then we propose a new update policy that is particularly appropriate to the sparse condition. In a first part grid based representations and update are presented. In a second part update policies are studied. Then in the third section is discussed how the new update policy modified the hierarchical algorithms for sparse OGs presented in our previous works. Finally results are presented in the last section.

1.2 Grid-based maps

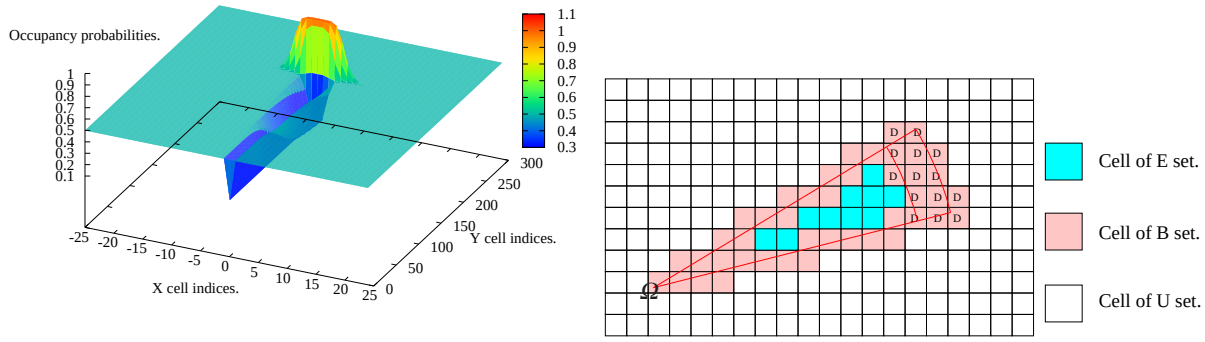
In this section the process of information fusion in grid-based map is described.

1.2.1 Occupancy grids

Grid-based representations were first introduced by Elfes and Moravec with the occupancy grids (OGs): in each cell of the grid a probabilistic estimation is maintained for the presence of an obstacle. This representation is closely related to range sensors for which sensor models exists that take into account measurement uncertainty to update the map.

Sensor model

- Z a random variable[1] for the sensor range measurements in the set $\mathscr{Z}$.
- $O_{x,y} \in \mathscr{O} \equiv \{\text{occ}, \text{emp}\}$. $O_{x,y}$ is the state of the cell (x,y), where $(x,y) \in \mathbb{Z}^2$. $\mathbb{Z}^2$ is the set of indexes of all the cells in the monitored area.
- the OGs needs three probability distribution to be defined: the *a priori* map occupancy $P_0(O_{x,y})$ and the two sensor models: $P(Z|[O_{x,y} = \text{occ}])$ and $P(Z|[O_{x,y} = \text{emp}])$. The two last modelling how uncertain a measurement is with respect to the obstacle location (fig. 1(a)).



(a) (b)

Fig. 1.1. (a) Update of a 2D OG after a sensor reading, initially each cell occupancy was unknown, *i.e.* 0.5 probability. The sensor beam has an aperture of 7 degrees. The sensor is positioned in (0,0). It thus is the just adding in each cell the value of the log-ratio of the two sensor models. (b) A range-finder beam. The range finder is located at Ω and its field of view is surrounded by red boundaries. It defines the three kind of cell types, unknown cells (U set) in white, empty cells (E set) in light blue and occupied cells: the one close to an obstacle marked with a "D" (B set).

Log-ratio form of occupancy update

As the occupancy is a binary variable, a quotient between the likelihoods of the two states of the variable is sufficient to describe the binary distribution. The new representation used is:

$$\text{odd}(O_{x,y}) = \log \frac{p([O_{x,y} = \text{occ}])}{p([O_{x,y} = \text{emp}])} \tag{1.1}$$

In the Bayesian update of the occupancy, the quotient makes the marginalization term disappear and thanks to a logarithm transformation, sums are sufficient for the inference:

$$\log \frac{p(\text{occ}|z)}{p(\text{emp}|z)} = \log \frac{p(\text{occ})}{p(\text{emp})} + \log \frac{p(z|\text{occ})}{p(z|\text{emp})} = odd_0 + odd(z) \tag{1.2}$$

where odd_0 named the initial estimation of the cell occupancy is log-ratio.

[1] For a certain variable V we will note in upper case the variable, in lower case v its realization, and we will note $p(v)$ for $P([V = v])$ the probability of a realization of the variable.

1.2.2 Reflection probability maps

Reflection probability maps were described by Burgard and instead of the occupancy probability the map measures the probability of a reflection per grid cell and have the advantage of not requiring an explicit sensor model. For each cell two numbers are stored: the number of times a beam has traversed the cell (♯misses) and number of times a beam has ended in the cell (♯hits). Then the probability of reflection is estimated as:

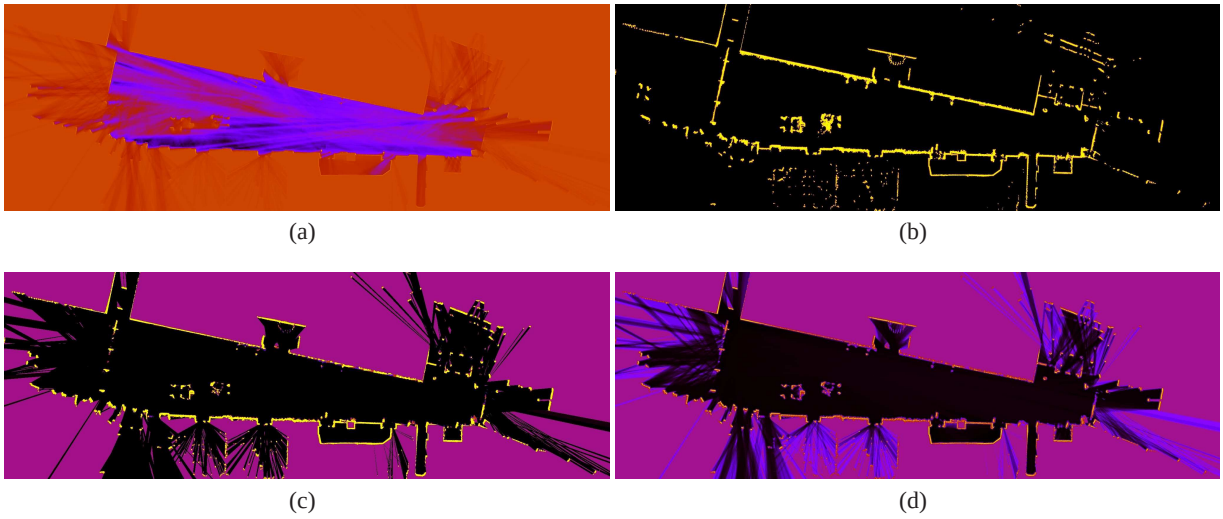
$$p(O_{x,y}) = \frac{\sharp\text{hits}}{\sharp\text{hits} + \sharp\text{misses}} \tag{1.3}$$

1.2.3 Normal transforms

The normal transforms store in addition of one of the previous kind of occupancy information the mean and the standard deviation of all the end-points of the beams that fall into the cell[2]. This representation was proposed by Biber in [1] and as a nice feature it reduces the problem of space discretization induced by the grid which greatly improved the localisation accuracy for instance.

1.3 Update Policies

In this section is studied the effect of long term sensor fusion on grid-based maps. In the first subsection the effect of the simple sensor fusion updates of the previous section are described highlighting the drawbacks. Then the usual max and clamping policies used to counter these drawbacks are reviewed. Last we present the policy we used along with the occupancy *a priory* that gives the best memory saving.



(a) (b)

(c) (d)

Fig. 1.2. Log-ratio occupancy maps from the Freiburg data set generated with various policies. For all the figure the color code is the following: the more bright is the color the more occupied is the cell. All color scales are defined from the min occupancy to the max occupancy of the maps. (a) No policy. (b) Max policy. (c) Clamping policy. (d) Exponential forgetting policy.

[2] Note that it supposed to store the number of time the cell was a beam end-point

1.3.1 No policy

When updates are performed using just the equation of the previous section, the map obtained suffers from two drawbacks: it is overconfident and it is highly irregular (fig. 2(a), fig. 3(a)). If carried in log-ratio space these two problems are evident through the unbounded sums along fusion updates. The number of observations required to change the state of a static cell is equal to the number of observations in the past that have defined this state. And if the robot have stayed a long time observing a part of the map, it will make a long time to change its state which is annoying since obstacles like car can be parked a long time before going quickly. This issue is called the map *overconfidence* problem. It can be understood as a memory problem since the map remember all agreeing observations. In particular as the vantage points of the sensor observations changes, all the minor changes in the pose of the sensor let a print in the map leading to very irregular representation. The term irregular means, here, that the function representing the map is far from a continuous function. And as there is no way to predict the patterns that appear in the map there is no way neither to compress the map efficiently.
However this update as a clear advantage: all the information are used to estimate the static parts of the map which made the location of the wall for instance very accurate (fig. 3(a)). It is worth notice that all these drawbacks and advantages are valid for all the grid-based representations.

1.3.2 Max policy

A solution, usually use because it is conservative ([8] or [12]) is to update using the maximum rule:

$$odd_t = \max(odd_{t-1}, odd(z)). \tag{1.4}$$

In this case, there is not any overconfidence problem for empty regions but an occupied area can never be empty anymore, the model cannot represent accurately the obstacles in case of false measurement or moving objects for instance. Furthermore the estimation of the obstacle position is very poor (thick wall in fig. 3(b)) which tends to give bad localisation. However the compression capabilities of such representation are interesting since most parts of the map stay at the original value when the space is empty (fig. 2(b)). For reflection probability maps the same effect can be obtained by just updating $\sharp$hits.

1.3.3 Clamping policy

A quantification policy discretized the range of occupancy values into a small set of possible values, like in [7]. This process is often done *a posteriori* since it is not obvious how to consistently update quantified values. For instance the minimum and maximum probabilities are not anymore respectively equivalent to 0 and 1 probabilities. The same policy is difficult to obtain with reflection probability maps since bounding $\sharp$hits and $\sharp$misses independently leads to wrong estimations. For log-ratio occupancy maps the update equation becomes:

$$odd_t = \max(\min(odd_{t-1} + odd(z), odd_{\max}), odd_{\min}). \tag{1.5}$$

where $\text{odd}_{\max}$ and $\text{odd}_{\min}$ are the predefined maximum and minimum log-ratio occupancy probabilities.

This policy is illustrated in fig. 2(c) and fig. 3(c). It is obvious that this policy suffers for the same problem of precision than the max policy. However, the state of the grid is reversible which make it more suitable for map updates. The regularity of the maps is good because most areas tend to converge towards the two extreme values.

1.3.4 Exponential forgetting policy

We present here a new policy that solve the precision, the overconfident and the regularity problem altogether. The update equations are the following:

$$odd_t = (1-\gamma)odd_{t-1} + \gamma odd(z). \tag{1.6}$$

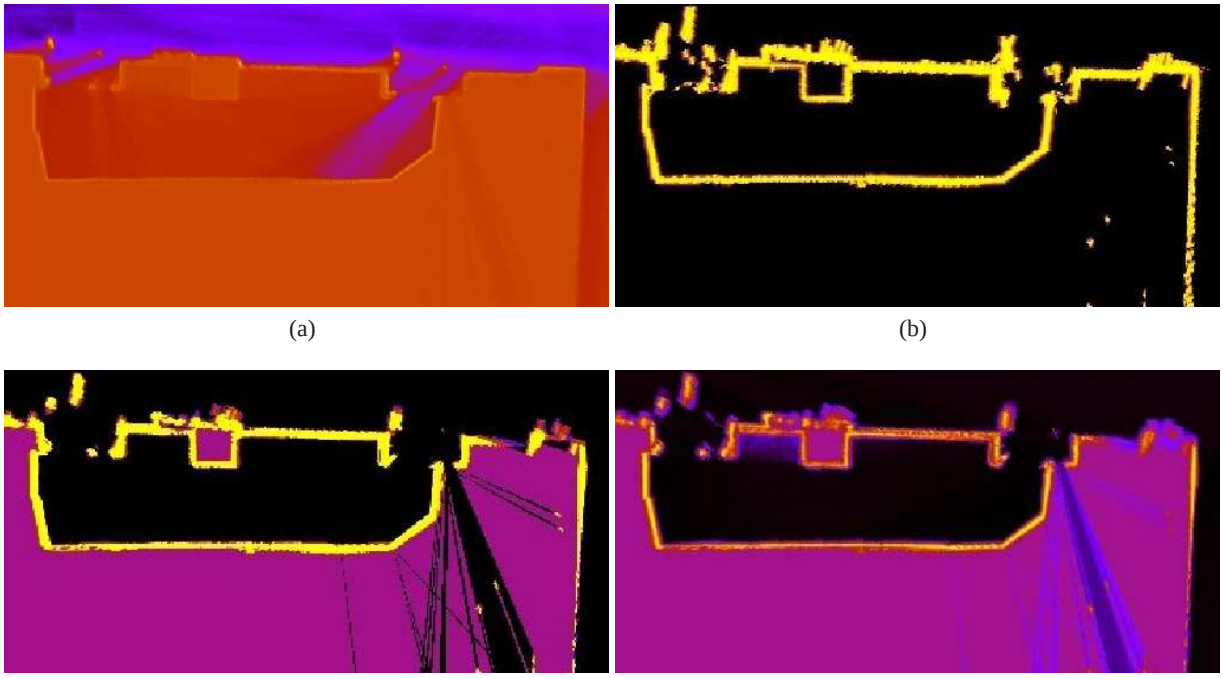


Fig. 1.3. Zoom on the map from the Freiburg data set generated with various policies. (a) No policy. (b) Max policy. (c) Clamping policy. (d) Exponential forgetting policy.

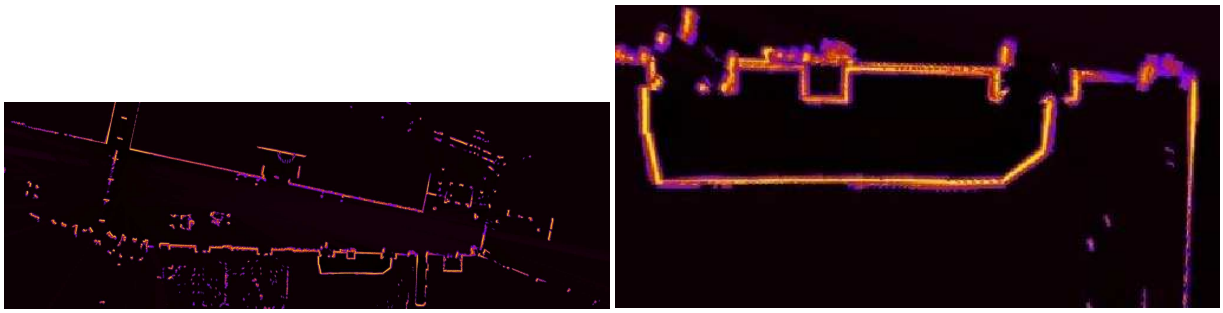
The map is naturally bounded, since the $odd(z)$ is bounded and if $odd_{t-1} = odd(z)$ then $odd_t = odd_{t-1}$. Therefore the map is easy to compress since there are large coherent areas (fig. 2(d)) and is not overconfident. The map keeps precision since the peaks of occupancy do not disappear because neighboring cells do not all reach the occupancy bounds but always keep their relative differences by forgetting (fig. 3(d)). These policy is called exponential forgetting since a past observation is exponentially less important in the current occupancy estimation. It is worth to notice that $\sharp$hits, $\sharp$misses, mean and standard deviation can be updated the same way.
The choice of γ determines the rate of forgetting and by trials we find that a low value is interesting in the case of the mapping with a laser-range finder at $10Hz$ ($\gamma = 0.3$ is taken in the experiments).

One drawback of this policy is that using naively in the unobserved area, it makes the cell occupancies converge toward a unknown state rapidly since in log-ratio the value of $odd(z)$ for this kind of area is zero. This problem can be avoided by only updating areas that are in the field of view of the sensor, in the occupied and empty zones only (see fig. 1(b))

1.3.5 Policy mixing and initial conditions

All these different policies can be combined together to build new policies that solve some of the problems. For instance, since the empty space is often the most important part of a map, an interesting combination is to clamp the lower occupancy values while not bounding the upper ones. The compression abilities of such a map are good, and it was the policy we use in previous experiments [16]. In this case, an obstacle can easily appear but it disappears very slowly if it was observed a long time. It is conservative from an obstacle avoidance point of view while it could be really annoying when searching a free place in a car park.
An other important choice is the initialization of the occupancy value of the map. From a compression point of view

the most interesting choice is to initialize the map with the empty bound[3] since it is the most probable value in an environment where a robot is supposed to evolve (see fig. 4(a) and fig. 4(b)).



(a) (b)

Fig. 1.4. Map from the Freiburg data set generated with the exponential forgetting policy and a initialization to an empty map. (a) Entire map (b) Zoom on the map.

1.4 Sparse representation through hierarchical map organization

This section describe how the update policies change the hierarchical update algorithm developed in [16] for sparse representations.

1.4.1 Problem statement

One popular method to compress a grid-based representation in the robotic community rely on building a tree based hierarchical representation of the map ([7], [9], [10], [8], [16]). The idea is to represent large area with the same occupancy with a single node at an appropriate coarse scale. To be efficient these strategies suppose that the map have indeed large uniform occupancy area, therefore a policy that bound the map is always required at least for the main part of the map (which is the empty one). As stated in [15], it is often interesting to store differences between the scales instead of the plain values because small differences are the clue to notice uniform areas or a variant of this scheme like the Haar wavelet transform.
The problem in such representations is how to efficiently update the hierarchical representation taking into account sensor model uncertainties. In a previous work [16] we derive a hierarchical algorithm that allow to make efficient hierarchical updates with a clamping policy.and in particular updates that are hierarchical in the sens that large part of the map can be updated at large scale when a large part is discover as empty. That is very important in the car park situation when a car can occupied many cells at fine resolution for instance. The drawback of clamping-based policy is that they are either lacking in precision, the drawback when not bounding the occupancy is that the map is overconfident. In the following is exposed the modifications to bring to the hierarchical algorithm in order to use exponential forgetting policies.

1.4.2 hierarchical algorithm

The key idea in the hierarchical exploration of the grid space is to define a predicate: *existIntersection* that is true if a given set of grid cells intersect the volume defined by the field of view of the sensor beams (blue plus red cells in

[3] It is rarely done without any policy that bound the map since there is no really empty value, and the maps are often initialized with a uniform prior in those cases

fig. 1(b)). The absence of intersection indicates that the given set of cells are outside the sensor field of view and don't need updating. For the case of *existIntersection* evaluating to true, a special sub case would be when the set of cells are totally included in the sensor field of view, then all the cells of the set belong to E (blue cells in fig. 1(b)) and their occupancy are decreased by the same amount of odd_{emp}, eq. 1.2. As the algorithm is able to detect uniform regions recursively, the grid representation should allow the update of regions, and wavelets provide a natural mechanism for doing so. For each grid area, the *existIntersection* predicate guides the search. If there is intersection the traversal searches deeper into the grid hierarchy, *i.e.* exploring finer scales. Otherwise it stops at the current node.

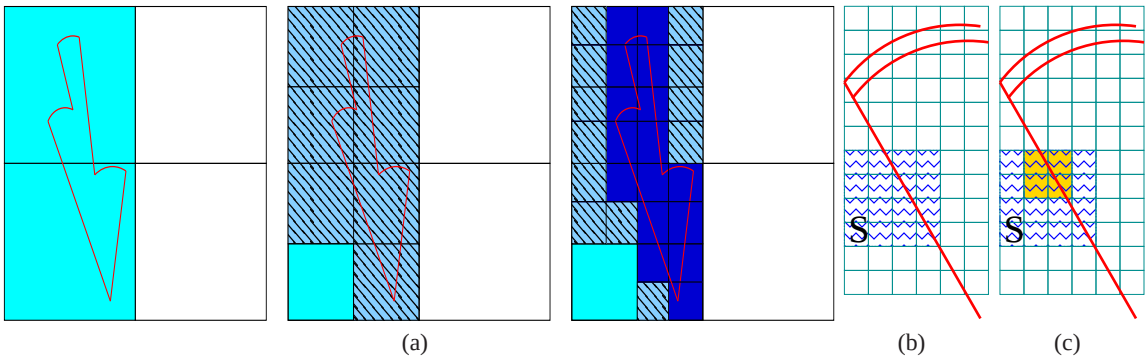
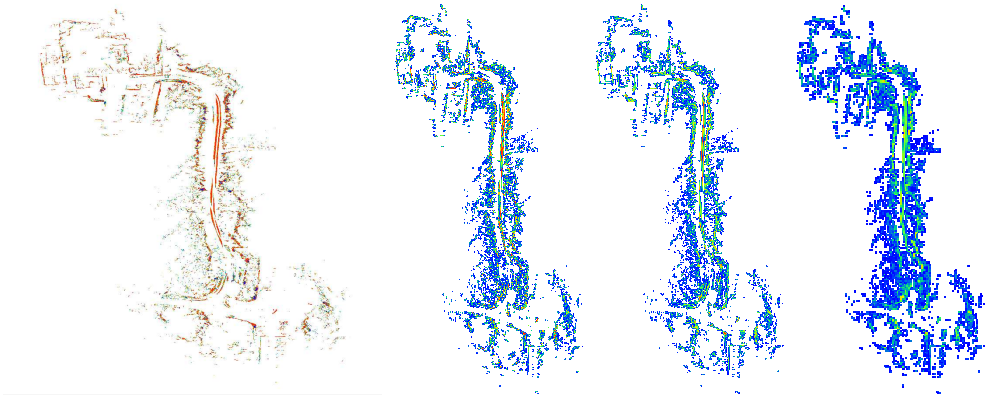


Fig. 1.5. (a) The hierarchical process of updating the grid: from the coarsest scale to the finest. To save computing time, area that are outside the polygon of view or totally included inside the area classified as empty are detected and processed early in the hierarchy. (b) and (c): two different cases for the iteration along a boundary of the field of view that separates E set and U set. Fig. 5(b) artificial separation, S (with waves) was totally empty and the observation of a part of its interior (on the right of the red boundary) does not bring any information gain. Fig. 5(c) the separation brings information about the state of the yellow area that is inside the field of view (on the right of the red boundary). Just in this last case the algorithm 2 performs a search at a finer scale.

An update is almost always required for cells that are in the obstacle neighborhood (cells marked with 'D' in fig. 1(b)) so iteration is always performed in area that contains such a cell. But for boundaries that separate cells that belongs to U set and to E set (white and blue cells in fig. 1(b)) iteration is required only if the E set corrects the knowledge in the grid (fig. 5(c)) otherwise the iterations can stop early in the hierarchy (fig. 5(b)).
Further details and pseudo code can be found in [16].

1.4.3 Modification of the update algorithm under exponential forgetting update policy

The main contribution of the update algorithm with the wavelet representation is that for uniform area the update is hierarchical, however with the exponential forgetting policy all finer scales must forget too. This an important drawback of this policy for a hierarchical representation.
To solve this issue there is the possibility to always proceed to a complete traversing of the tree except for regions outside the sensor field-of-view (U set) or regions with occupancy that has reached one of the boundaries occupancy values. The solution proposed here improves this by noticing that a hierarchical update is worth when a big part of the environment being previously occupied becomes empty (like with parked car that leaves its place), in that case successive updates will lead to a complete empty area. Therefore updating the grid at fine scale is totally useless and time consuming, since big parts will be empty.
We propose to keep track of the empty updates performed at a coarse scale by recording the corresponding coarse nodes of the grid in a second tree T_e. In that tree is stored the corresponding nodes from which empty updates are required along with the number of time an empty update was performed. When updating the grid tree, T_e is traversed too and if an update is required for child of a node in T_e the empty updates are performed in a lazy fashion. The equations are the following:



(a) (b) (c) (d)

Fig. 1.6. Results of algorithm 2 for 2D data: scaled views of the OG provided by different scales of WavOG representation (a) scale 0, cell size: $10cm \times 10cm$. (b) scale 2, cell size: $40cm \times 40cm$. (c) scale 3, cell size: $80cm \times 80cm$. (d) scale 4, cell size: $1.60m \times 1.60m$. The data where provided by CSIRO on a long-run experiment. The size of the map is approximately $450m \times 200m$. The number of cells vary from 9 Millions to 36000 from scale 0 to scale 4 and the ratio is one for 256 if the scale 4 is compared to the scale 0. It is obvious that the precision of the map decreases when the scale increases however the shape of the environment is sufficient enough for path planning since scale 3 in the big open area which is very interesting and promising for multiscale path planning algorithm for instance.

$$\begin{aligned} odd_t &= (1-\gamma)^n odd_{t-1} + \gamma \mathrm{odd}_{\mathrm{emp}} (\sum_{i=0}^{n-1} (1-\gamma)^i), \\ &= (1-\gamma)^n (odd_{t-1} - \mathrm{odd}_{\mathrm{emp}}) + \mathrm{odd}_{\mathrm{emp}}. \end{aligned} \tag{1.7}$$

when n is the number of cached empty update stored in the coarse node of T_e. T_e is then updated by zeroing the cached updates at the coarse node. It the empty update is required for a fine scale the whole tree is traversed. The only parameter to set is the threshold between fine scale and coarse scale which we fix to match half the size of a car in our application.

1.5 Experiments

1.5.1 Computing time and required memory

We performed experiments[4] on 2D real data with the modified hierarchical algorithm for exponential forgetting policy with moving obstacles.

In the 2D experiment a big truck equipped with four SICK LMS-291 at each corner carries a big hot metal container behind it during a 1.5 hours experiment, the data are noisy and the evolution property of the map is required by the presence of a lot of moving obstacles (in particular the hot metal container). The algorithm processes the 4 laser range-finder at real-time ($40Hz$ each). The processing time of the modified algorithm matches the processing time of the update algorithm with clamping policy but provides in addition a non overconfident map.

[4] Every experiment was done with an Intel(R) Pentium(R) 4 CPU $3.00GHz$.

1.6 Conclusions and future works

1.6.1 Conclusions

The objective of this work is to present a review of the update policies in grid-based representation that makes this kind of environment modelling more suitable for compression and to present a new one based on exponential forgetting. This new update policy offers many advantages compare to the others because it present neither overconfidence nor loss of accuracy in the long term. However it remained to see how this new update pattern fit with our hierarchical update algorithm of a sparse grid representation. We have then presented a scheme that allows lazzy updates in case of gross disapearance of objects like cars that leave a car park. In that case the algorithm delayed update as long as possible and in most of the case never needs to do them which makes it as efficient than the update algorithm with Haar wavelets and clamping update policy.

1.6.2 Future Works

In the future we will explore parallelising the computing of the hierarchical update algorithm since the tree structure and distributed memory organisation of the data structure seems be well designed for such a purpose.

1.7 Acknownledgments

The authors would like to thank Christopher Tay Meng Keat, Christophe Braillon and Cedric Pradalier for their contributions and fruitful discussions. We express also our gratitude to CSIRO for providing a consistent huge data set that allowed us to validate our method. We also thank Cyrill Stachniss for making corrected data sets available through his web page [11].

References

1. Peter Biber and Wolfgang Straßer. The normal distributions transform: A new approach to laser scan matching. In *IEEE/RJS International Conference on Intelligent Robots and Systems*, 2003.
2. D. Cole and P. Newman. Using laser range data for 3d slam in outdoor environments. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, Florida, 2006.
3. Alberto Elfes. *Occupancy grids: a probabilistic framework for robot perception and navigation*. PhD thesis, Carnegie Mellon University, 1989.
4. Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981.
5. G. Grisetti, C. Stachniss, and W. Burgard. Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2443–2448, 2005.
6. Zia Khan, Tucker Balch, and Frank Dellaert. Mcmc data association and sparse factorization updating for real time multitarget tracking with merged and multiple measurements. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(12):1960–1972, 2006.
7. Gerhard K. Kraetzschmar, Guillem Pagès Gassull, and Klaus Uhl. Probabilistic quadtrees for variable-resolution mapping of large environments. In M. I. Ribeiro and J. Santos Victor, editors, *Proceedings of the 5th IFAC/EURON Symposium on Intelligent Autonomous Vehicles*, July 2004.
8. Pierre Payeur, Patrick Hébert, Denis Laurendeau, and Clément Gosselin. Probabilistic octree modeling of a 3-d dynamic environment. In *Proc. IEEE ICRA 97*, pages 1289–1296, Albuquerque, NM, Apr. 20-25 1997.
9. Pierre Payeur, Denis Laurendeau, and Clément Gosselin. Merging uncertainty into probabilistic octree models of 3-d perturbed workspaces. In *Proc. VI-98*, pages 439–446, Vancouver BC, June 18-20 1998.
10. Pierre Payeur, Denis Laurendeau, and Clément Gosselin. Range data merging for probabilistic octree modeling of 3-d workspaces. In *Proc. ICRA 1998*, pages 3071–3078, Leuven, Belgium, May 16-21 1998.
11. Cyrill Stachniss. Corrected robotic log-files. http://www.informatik.uni-freiburg.de/ stachnis/datasets.html.

12. Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*, chapter 9, page 231. The MIT Press, 2005.
13. Rudolph Triebel, Patrick Pfaff, and Wolfram Burgard. Multi-level surface maps for outdoor terrain mapping and loop closing. In *"Proc. of the International Conference on Intelligent Robots and Systems (IROS)"*, 2006.
14. J. Vandorpe, H. Van Brussel, and H. Xu. Exact dynamic map building for a mobile robot using geometrical primitives produced by a 2d range finder. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 901–908, Apr 22-28 1996.
15. M. Yguel, O. Aycard, and C. Laugier. Wavelet occupancy grids: a method for compact map building. In *Proc. of the Int. Conf. on Field and Service Robotics*, 2005.
16. M. Yguel, C. Tay Meng Keat, C. Braillon, C. Laugier, and O. Aycard. Dense mapping for range sensors: Efficient algorithms and sparse representations. In *Proceedings of Robotics: Science and Systems*, Atlanta, GA, USA, June 2007.